

2207/7083
PATENT

UNITED STATES PATENT APPLICATION
FOR

**METHOD AND PROCESSOR FOR RECOVERING REGISTERS FOR REGISTER
RENAMING STRUCTURE**

INVENTORS:

Stephan J. Jourdan

Michael Bekerman

Ronny Ronen

PREPARED BY:

KENYON & KENYON
1500 K ST., N.W.
WASHINGTON, D.C. 20005
(202) 220-4200

METHOD AND PROCESSOR FOR RECOVERING REGISTERS FOR REGISTER RENAMING STRUCTURE

Technical Field

5 The present invention relates to executing instructions in a processor. More particularly, the present invention relates to reclaiming physical registers in a register renaming structure for out-of-order instruction execution.

Background of the Invention

10 Most instructions in a computer instruction set operate on several source operands to generate results. The instructions name, either explicitly or through an indirection, the source and destination locations where values are read from or written to. A name may be either a logical, or an architectural, register or a location in memory.

15 Instructions involving register operands are faster than those involving memory operands. For some microprocessor architectures, instructions naming memory operands are translated, or decoded, into microinstructions that transfer operand values from memory to logical registers and then perform the decoded computations. The number of logical registers often is limited, compilers efficiently should utilize logical registers to generate efficient code.

20 The number of physical registers available in a microprocessor typically exceeds the number of logical, or architectural, registers, so that register renaming may be utilized to increase performance. More than one physical register may be renamed to a logical register. Each physical register may correlate to a non-retired instruction, and, therefore, a plurality of non-retired instructions should utilize a plurality of physical registers. In particular, for out-of-order processors, register renaming allows instructions to be executed out of their original program order. Thus, for
25 many out-of-order processors, an instruction is renamed so that logical registers named in the original instruction are renamed to physical registers.

Renaming a logical register involves mapping a logical register to a physical register. These mappings are stored in a Register Alias Table ("RAT"). A RAT maintains the latest mapping for each logical register. A RAT is indexed by logical registers, and provides mappings to corresponding physical registers. This activity may be called dependency tracking.

Fig. 1 depicts a register renaming and dependency tracking scheme involving three structures: a RAT 110, a active list 102, and a free list 104. For each logical register specified by a renamed instruction, an unused physical register from the free list 104 is allocated. RAT 110 is updated with this new allocation. Physical registers are free to be used again, or reclaimed, once they cannot be referenced by instructions in the current instruction window.

Based upon the data structures depicted in Fig. 1, one method for register reclaiming is to reclaim a physical register when the instruction that evicted it from RAT 110 retires. Thus, the instruction that created the new allocation to the physical register is retired for reclaiming to occur. As a result, whenever a new allocation updates RAT 110, the evicted old allocation is pushed into active list 102. An active list 102 entry is associated with each instruction in the instruction window. When an instruction retires, the physical register evicted from RAT 110 and recorded in active list 102, if any, is reclaimed and pushed into free list 104. The cycle is depicted in Fig. 1.

During register renaming, free list 104 allocates an unused physical register when an instruction specifies a logical, or architectural, register as a destination. The instruction source physical registers are read from RAT 110 based on the instruction source logical registers. Afterwards, RAT 110 maps the destination logical register to the newly allocated physical register from free list 104. When a destination logical register is renamed, subsequent instructions cannot read the physical register that previously was mapped to that logical register. As discussed above, an appropriate condition for register reclaiming is to reclaim a physical register when the instruction that generated the new mapping in RAT 110 retires. Further, the old mapping is pushed into active list 102 from RAT 110. When the corresponding instruction retires, the old mapping is reclaimed and pushed into free list 104.

Fig. 2 depicts an example of a recovery scheme for the register renaming structure. The renaming structures described in Fig. 1 are repaired when an exception occurs in the processor. Recovery techniques seek to restore the correct state to RAT 110, active list 102 and free list 104. Several events may terminate a normal instruction flow. Common reasons for exceptions include

mispredicted branches, interrupts, or exceptions. When these events occur, instruction window buffer 112 should be flushed after the point of the exception. In addition, all three structures involved in register renaming should be repaired to reflect the new logical, or architectural state.

Fig. 2a depicts the original state of the processor. Instruction window 110 holds four instructions that have not been renamed. Instructions 1, 3 and 4 specify logical register EAX as their destinations. Active list 102 is empty. The current mapping of logical register EAX is physical register R1. Free list 104 contains unallocated registers R2 to R5.

Fig. 2b depicts that state of the renaming structures after all four instructions are renamed. Each allocation of a new physical register for EAX pushed the previous mapping into active list 102. The current mapping of EAX in RAT 110 is physical register R4. Physical registers R1, R2 and R3 correlate to old mappings evicted from RAT 110. Free list 104 contains unallocated physical register R5.

If the branch instruction of instruction 2 in instruction window buffer 112 is mispredicted, then all three remaining structures need to be repaired so they reflect the state of the machine right after instruction 2. One method to recover RAT 110 is to establish checkpoints of its content for every dispatched branch. RAT 110 can be recovered by copying the content of the corresponding checkpoint into RAT 110. Active list 102 may be repaired by flushing all entries past the mispredicted branch, or instructions 3 and 4 in the example. Fig. 2c depicts a complex algorithm to recover free list 104. Part of the physical registers in RAT 110 and active list 102 are returned to free list 104. Algorithms may need to be implemented to recover the physical registers to free list 104. Merely recovering flushed entries may lead to physical registers being recovered incorrectly. Thus, a need has arisen for a method for recovering free list 104 in a more efficient manner.

Brief Description of the Drawings

Fig. 1 illustrates a register renaming structure for a processor.

Figs. 2a-c illustrate an example of a recovery scheme for a processor.

Fig. 3 illustrates a block diagram of a processor in accordance with an embodiment of the present invention.

Fig. 4 illustrates a block diagram of a register renaming structure in accordance with an embodiment of the present invention.

Fig. 5 illustrates a flowchart of a method for recovering registers in accordance with an embodiment of the present invention.

Fig. 6 illustrates a flowchart of a method for recovering registers in accordance with an embodiment of the present invention.

Detailed Description

An embodiment of the present invention is directed to a processor having a register renaming structure to recover a free list. The processor may include a physical register file having physical registers. The processor also may include a decoder to decode an instruction indicating a destination logical register. The processor also includes a register allocation table to map the destination logical register to an allocated physical register. The processor also includes an active list that includes an old field and a new field. The old field includes at least one evicted physical register from the register alias table. The new field includes the allocated physical register. The processor also includes the free list of unallocated registers reclaimed from the active list.

Fig. 3 is a block diagram of a processor 200 in accordance with an embodiment of the present invention. Processor 200 may be an out-of-order processor. For simplicity, not all functional units of processor 200 are depicted. In processor 200, fetch unit 202 fetches instructions from instruction cache 204, and decode unit 206 decodes these instructions. For a Complex Instruction Set Computer ("CISC") architecture, decode unit 206 decodes a complex instruction into one or more micro-instructions. The micro-instructions may define a load-store type architecture, so that micro-instructions involving memory operations are simple load or store instructions. The present invention, however, may be practiced for other architectures, such as Reduced Instruction Set Computer ("RISC"), Very Large Instruction Word ("VLIW") architectures, and the like.

For a RISC architecture, instructions are not decoded into micro-instructions. Because the present invention may be practiced for RISC architectures as well as CISC architectures, we shall not make a distinction between instructions and micro-instructions unless otherwise stated, and will refer simply to these as "instructions".

A register renaming functional unit 208 may include register allocation table ("RAT") 210. RAT 210 contains current mappings between logical registers and physical registers. The physical registers are indicated by physical register file 212. The decoded instructions may indicate the logical, or architectural, registers to be utilized by the instruction. Every logical register should have a mapping to a physical register in physical register file 212. As noted above, more than one physical register may be mapped to a logical register. This number may be dependent upon the number of non-retired instructions. Some physical registers in physical register file 212 may be dedicated for integers whereas others may be dedicated for floating point numbers, but for simplicity, these distinctions are not made.

During renaming of an instruction, current entries in RAT 210 may provide the required mapping, or mappings, for renaming the source logical register, or registers, of the instruction, and a new mapping is created for the destination logical register of the instruction. The new mapping may evict the old mapping in RAT 210, and RAT 210 is updated with this new mapping.

Renamed instructions are placed in instruction window buffer 226. Entries in instruction window buffer 226 contain the opcode of the renamed instruction, as well as other fields. The results of the instructions, however, may not be stored in instruction window 226 but may reside in physical register file 212. Instruction window buffer 226 may allow for instruction retirement in original program order. Instruction window buffer 226 may operate as a circular buffer, where a first pointer points to the next instruction to be retired and a second pointer points to the next available entry for a newly decoded instruction. When an instruction retires, the first pointer is incremented. When a decoded instruction is placed in instruction window buffer 226, the second pointer is incremented. The pointers are incremented by one for each decoded instruction in a RISC architecture, and may be incremented by more than one for CISC architectures.

According to embodiments of the present invention, instruction window buffer 226 may have pointer fields for pointing to physical registers. When an instruction placed in instruction window buffer 226 has evicted a mapping in RAT 210, a pointer field associated with the instruction is set to point to the physical register of the evicted mapping. The pointer fields comprise an active list 216. Active list 216 includes old field 218 and new field 220. Free list 214 may be a queue of pointers pointing to unallocated physical registers. Free list 214 also may reclaim allocated registers from active list 216 as they become available.

Active list 216 may have two fields. Old field 218 stores the evicted registers from RAT 210. This process is similar to active list 102 depicted in Fig. 1. New field 220 stores, for each entry, the physical register newly allocated in RAT 210. Thus, active list 216 may record the allocated physical register from free list 214 for each instruction. Upon a misprediction or other exception within processor 200, free list 214 may be repaired by pushing back all physical registers pointed by new field 220 of all instructions past the mispredicted branch. Thus, in addition to active list 216 having evicted physical registers in old field 218, active list 216 also has the newly allocated physical registers in new field 220. Therefore, free list 214 may reclaim registers from active list 216 without algorithms or indirect processes executed at a misprediction condition.

Fig. 4 depicts an example of a register renaming structure 300 in accordance with an embodiment of the present invention. Register renaming structure 300 may be analogous to renaming structure 208 depicted in Fig. 3. RAT 302 has a current mapping of logical register EAX to physical register R5. Instruction window buffer 320 illustrates five instructions that have been renamed. Active list 306 includes old field 308, new field 310 and bit field 312. Free list 304 indicates that physical registers R6 and R7 are unallocated. In instruction window buffer 320 indicates that instructions 1, 2, 4 and 5 specify logical register EAX as their destination.

Each allocation of a new physical register for logical register EAX may push the previous mapping by RAT 302 into active list 306, and stored in old field 308. New field 310 indicates the newly allocated physical register by RAT 302 for each instruction. Because instruction 3 is a branch, a physical register should not be allocated by RAT 302. Thus, active list 306 should not store any physical register information for instruction 3. Thus, according to the embodiment depicted in Fig. 4, all five instructions have been renamed, with physical registers R1-R4 having been evicted from RAT 302, as depicted by old field 308.

New field 310 may contain the newly mapped physical register by the RAT 302 for each renamed instruction. For example, instruction 1 specifies logical register EAX as its destination physical register. RAT 302 evicts physical register R1 to old field 308 and maps logical register EAX for instruction 1 to physical register R2. This mapping is recorded in new field 310. As additional instructions are renamed, old field 308 and new field 310 record the evicted physical register and newly allocated, or mapped, physical register, respectively.

A misprediction condition may occur that impacts register renaming structure 300. For example, instruction 3 may be a branch instruction having a predicted result. If the prediction is in

error, then the branch is “mispredicted” and a misprediction condition occurs. Instructions executed after the misprediction are flushed because they are incorrect. Thus, referring to instruction window buffer 320, instructions 4 and 5 are flushed and the register renaming structures placed back to their states at the misprediction.

5 As noted above with reference to Fig. 2, RAT 302 may have snapshots taken at each checkpoint. The snapshot then may be used to restore RAT 302 to its states prior to the branch. Other schemes may exist to perform this process. Active list 306 may be restored by flushing all entries past the misprediction. Thus, for example, entries allocated after the mispredicted branch within old field 308 and new field 310 may be flushed out of active list 306. Entries associated with
10 instructions 3 and 4 are flushed. Free list 304 should recover those physical registers freed up by RAT 302 and active list 306. According to embodiments of the present invention, free list 304 can recover physical registers by pushing back all physical registers that are indicated in new field 310 past the misprediction condition. Physical registers R4 and R5 are to be recovered by free list 304.

According to an embodiment of the present invention, free list 304 may recover physical registers by using the normal process of reclaiming physical registers at retirement time. As noted above, physical registers indicated by old field 308 in active list 306 are reclaimed whenever an instruction retires. Embodiments of the present invention make use of the existing reclaiming mechanism by having it to work differently for a recovery after a misprediction. Retirement pointer 322, which usually points to the next instruction to retire in both the instruction window 320 and the active list 300, is moved to the first wrong instruction in instruction window buffer 320. All instructions belonging to the mispredicted path are “retired,” but instead of reclaiming physical registers evicted by the associated instruction from old field 308, physical registers indicated by new field 310 are reclaimed. This procedure may be known as a “dummy” retirement and is performed for resource reclaiming. Retirement may be halted for during the dummy retirement. Dummy retirement has no effect except to recover physical registers after a misprediction condition. Because newly dispatched instructions overwrite entries associated to the mispredicted path, the dummy retirement rate should match the dispatch rate of new instructions. The recovery process may be performed without impairing performance because room in instruction window buffer 320 is created by the flushing operations caused by the misprediction condition. Thus, overriding entries on active list 306 will not impair performance of register renaming unit 300. Therefore, recovering of physical registers is performed on the evicted physical registers of old field 308 when retiring instructions,

whereas recovering of physical registers is performed on the newly allocated physical registers when recovering from an exception.

According to another embodiment of the present invention, physical register recovery for free list 304 may occur during dispatch of the instructions after the register are renamed. Dispatch may be defined as the process of entering instructions in instruction window 320. This embodiment uses bit field 312 within active list 306. Bit field 312 includes bits associated with each entry of active list 306. Thus, in addition to old field 308 and new field 310, entries of active list 306 also will have a bit in bit field 312. Recovery is performed on the entry overwritten in active list 306 by the newly dispatched instruction. Active list 306 may be a circular buffer that continuously overwrites entries as instructions are dispatched. If the instruction associated with the entry in active list 306 was retired correctly, then the recovery, or reclaiming, is performed based on old field 308. If the instruction associated with the entry was not retired correctly, then the recovery is performed based on new field 310. This action indicated the entry corresponded to an entry that was flushed. An example of recovering a physical register on a non-retired instruction is when the instruction has been flushed from instruction window buffer 320 after a misprediction condition.

The bits within bit field 312 correlate as to how the instruction should be treated. For example, a zero ("0") bit may indicate that the instruction was retired correctly. A one ("1") may indicate that the instruction was not retired correctly, but flushed due to a misprediction. During a misprediction condition, active list 306 may set the bits after the misprediction instruction to a value of 1. Pointer 322 is aligned at the first entry after the branch instruction, and those entries afterwards are assigned a value of 1 in the active list 306. Thus, the entries associated with flushed instructions 4 and 5 have a bit value of 1 in bit field 312. As new instructions are dispatched, they are written into active list 306. Register renaming structure 300 looks at the bit in bit field. If the value is 0, then the physical register indicated in old field 308 is reclaimed by free list 304. If the value is 1, then the physical register indicated in new field 310 is reclaimed by free list 304. For example, with regard to instruction 4, after the misprediction condition, physical register R4 would be reclaimed by free list 304. Thus, at dispatch time, the bit is set in bit field 312, and the bit is cleared when the associated instruction retires.

Fig. 5 depicts a flowchart of a method for recovering registers in a processor in accordance with an embodiment of the present invention. Embodiments of the present invention may allocate physical registers to the RAT. [Box 502] For example, referring to Fig. 4, RAT 302 may be

allocated a physical register from free list 304. Embodiments of the present invention may move an evicted physical register from RAT 302 to old field 308 in active list 306. [Box 504] The newly allocated physical register may be moved to new field 310. Embodiments of the present invention may detect a misprediction condition correlating to an instruction associated with an entry on active list 304. [Box 506] A misprediction condition may be a mispredicted branch of a branch instruction. The results of the misprediction should be flushed from register renaming structure 300. Embodiments of the present invention may move pointer 322 to the next entry following the mispredicted branch. [Box 508]

Embodiments of the present invention may use register renaming structure 300 to overwrite entries in active list 306 based on the processes described above. [Box 510] Embodiments of the present invention may reclaim allocated physical registers in new field 310 to free list 304. [Box 512]

Fig. 6 depicts a flowchart of another method for recovering registers in a processor in accordance with another embodiment of the present intervention. Embodiments of the present invention may set a bit during a misprediction condition. [Box 602] For example, referring to Fig. 4, the bit in bit field 312 is set during a misprediction condition for entries after pointer 322, which has been moved back to the entry following the mispredicted branch.. Embodiments of the present invention may overwrite an entry in active list 306, including the physical registers in old field 308 and new field 310. [Box 604] Embodiments of the present invention may read a bit in bit field 312 in active list 306 and correlating to the overwritten entry. [Box 606] Embodiments of the present invention may determine whether the bit read in step 606 is equal to 0. [Box 608] If so, then embodiments of the present invention may reclaim the physical register indicated in old field 308. [Box 610] For example, if the entries correlating to instruction 4 are overwritten and instruction 4 was retired, then physical register R3 would be reclaimed. If not, then embodiments of the present invention may reclaim the physical register in new field 310 correlating to the instruction. [Box 612] Thus, if instruction 4 was flushed as a result of a misprediction, and the bit is 1, then physical register R4 would be reclaimed.

By reclaiming physical registers from the active list to the free list, embodiments of the present invention may avoid additional logic or algorithms for reconciling misprediction conditions. Further, physical registers may be reclaimed when the registers are not correlating to an instruction

that is not retired or flushed from the instruction window buffer. Thus, physical register resources may be utilized more efficiently and without increased risk.

Thus, it is apparent that there has been provided, in accordance with the embodiments of the present invention disclosed above, a method and apparatus recovering registers for a register
5 renaming structure. Although the present invention has been disclosed in detail, it should be understood that various changes, substitutions, and alterations may be made herein. Moreover, although software and hardware are described to control certain functions, such functions can be performed using either software, hardware or a combination of software and hardware, as is well known in the art. Other examples are readily ascertainable by one skilled in the art and may be made
10 without departing from the spirit and scope of the present invention as defined by the following claims.

11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211